

EISC 기반 블루투스 베이스밴드 SoC 설계 및 검증

*전 군 선, *김 현 미, **임 재 윤
*제주대학교 통신공학과, **제주대학교 통신컴퓨터공학부
전화 : 064-754-3635 / 핸드폰 : 017-690-7994

EISC based BlueTooth Baseband SoC Design and Verification

Gun Seon Jin, Hyun Mi Kim, Jea Yun Lim
Dept. of Telecommunication Engineering, Cheju National University
E-mail : mpeg99@hotmail.com

Abstract

To design the Bluetooth core efficiently, I analyzed the entire architecture and basic functions in the baseband and attempted to implement a Bluetooth one-chip solution on the basis of the Bluetooth SIG specification 1.1. We implemented important blocks into the hardware and firmware and found increased efficiency implementation when compared with the results of the implementation that using the criterion of size, performance, stability, etc.

And then to connect the baseband to the SE3208 core of the EISC type, we defined the baseband register and discovered a suitable method by comparing two results with the two connection ways.

I. 서론

블루투스는 이동전화, 컴퓨터, PDA등 근거리 무선접속을 쉽게 연결할 수 있게 해주는 기술로, 컴퓨터 및 통신 산업체의 규격이다. 블루투스는 10m이내의 범위 안에서 통신하는 것을 목표로 하며 출력에 따라 100m 까지 가능하다. 블루투스가 갖는 가장 큰 특징은 전방향 무선통신이고 차폐를 투과성이 있어 공간 활용도를

높이면서 휴대 정보통신 기기를 가방이나 주머니에 넣은 채로 다른 정보통신기기와 통신할 수 있다.

본 논문에서는 블루투스 칩을 효율적으로 설계하기 위하여 블루투스 SIG 스펙1.1을 기준으로, 베이스밴드의 전체적인 구조와 기본 기능을 분석하였고 신정한 MCU를 이용하여 CPU 내장 블루투스 원칩 솔루션 구현의 기반을 마련하고자 이를 시도하였다.

이를 위해 주요 블럭을 크기와 성능, 안정성 등을 토대로 하드웨어와 펌웨어로 구현하여 그 결과를 비교 검토함으로써 블록에 대한 효율적인 구현형태를 알아보았다. 그리고 나서 베이스밴드의 레지스터를 정의하고 이것과 함께 검증된 베이스밴드 블록을 EISC방식의 SE3208 코어와 두 가지 방식으로 연결하여 구현하였고 이를 FPGA로 검증하고, 그 결과를 비교 검토함으로써 적합한 방법을 제안하였다.

II. 블루투스 베이스밴드 구조 및 블록 구현

베이스밴드는 RF모듈을 제외하면 블루투스 스택에서 가장 하부에 위치한 계층으로, 일반적인 LAN을 예로들면 NIC정도에 해당한다. 베이스밴드는 가장 하부에서 블루투스 디바이스 간의 연결을 담당하고 그 구성 블록도는 그림1과 같다.

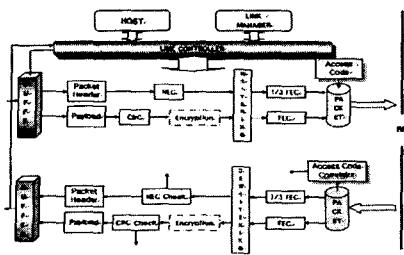


그림 1. 블루투스 베이스밴드 블럭도

1. Access code

Access code는 모든 패킷에 포함되어 패킷의 가장 처음에 전송되고, 패킷을 수신했을 때 가장 먼저 체크되는 부분이다. Access code는 프리앰블(preamble), 동기워드(syncword), 트레일러(trailer)로 구성되며 헤더가 있는 경우는 72비트의 길이를 갖고 없는 경우는 68비트의 길이를 갖는다.

그림2는 동기워드의 생성과정을 자세히 보여준다.

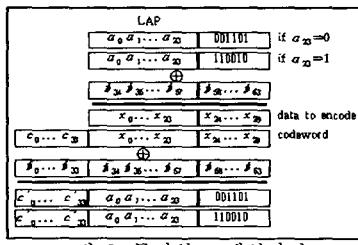


그림 2. 동기워드 생성과정

Access code는 블루투스 기기의 고유 주소 LAP가 '0x9E8B33'인 경우 테스트 한 결과를 나타내었다. 그림3과 4는 그림2의 Access code를 하드웨어와 펌웨어로 구현한 결과이다.



그림 3. Access code 하드웨어 시뮬레이션

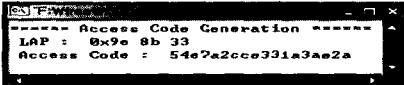


그림 4. Access code 펌웨어 시뮬레이션

Access code를 구현한 결과 그 크기가 하드웨어는 3000Gates, 펌웨어는 2Kbyte임을 알 수 있었다. Access code는 각 상태에 따라 다른 LAP를 이용하기 때문에 블루투스가 새로운 연결을 설립하기 위해서는 여러 번 Access code가 생성되지만 일단 새로운 연결 상태로 진입하면 같은 Access code 값을 유지하게 되므로 메모리에 저장해 두었다가 계속적으로 사용하면

된다. 따라서 크기 면에서도 유리한 펌웨어로 구현하는 것이 좋을 것이다.

2. Security

(1) Authentication

블루투스에 사용된 인증은 challenge-response 구조로 되어 있다. 여기서 상대편이 공유되는 비밀키를 알고 있는지 체크하기 위해 양방향 프로토콜이 사용되는 데, 이는 두 디바이스가 같은 키를 갖고 있는지, 인증 과정을 성공적으로 수행했는지를 체크하게 된다. 이러한 인증을 위한 함수는 그림5와 같고, 이 과정에서 생성된 ACO 값은 양쪽 디바이스에 저장되고 나중에 암호화키를 만드는데 사용된다.

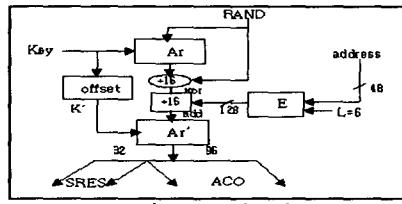


그림 5. E1 알고리즘

그림 6은 그림 5의 알고리즘을 펌웨어로 구현하여 입력 값이 다음과 같은 경우에 얻은 결과이다.

rand = bc3f30689647c8d7c5a03ca80a91eceb
address = 7ca89b233c2d
key = 159dd9f43fc3d328efba0cd8a861fa57

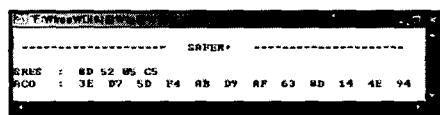


그림 6. E1 알고리즘의 펌웨어 시뮬레이션

E1 알고리즘은 순차적이고 메모리적인 요소가 많기 때문에 펌웨어가 하드웨어로의 구현 보다 구현 절차가 간소해 질 수 있다. 또한 이 블록은 두 디바이스간의 인증 절차에 사용되고 그 후 실제 정보 암호화는 다른 알고리즘을 사용하므로 펌웨어로 구현을 하는 것이 유리하며 그 구현 결과 크기가 6Kbyte임을 얻었다.

(2) Encryption

블루투스에서의 사용자 정보는 패킷의 페이로드 부분을 암호화함으로서 보호될 수 있다. 그림 7에 나타난 것처럼 블루투스 정보의 암호화는 모든 정보를 재동기화하는 E_0 라 불리는 Stream cipher에 의해 수행된다. 암호화 시스템은 세 단계로 구성되는데, 첫 단계에서는 초기화가 이루어지고, 두 번째 단계(key

stream generator)에서 생성된 key stream bit가 세 번째 단계에서 전송하고자 하는 데이터와 XOR되어 암호화된다.

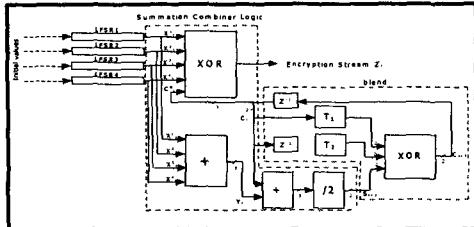


그림 7. 암호화기 알고리즘

그림8과 9은 다음의 입력값을 이용하여 암호화기 알고리즘을 구현한 결과이다.

$K'c = 00000000000000000000000000000000$
addr = 000000000000
clk = 00000003

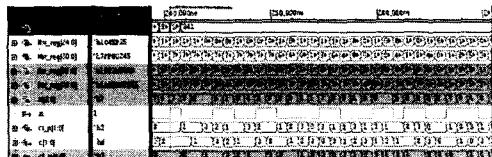


그림8 암호화기의 하드웨어 시뮬레이션

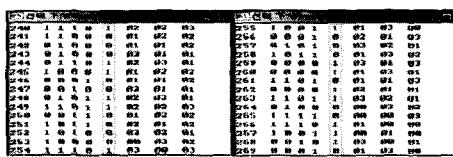


그림9 암호화기의 펌웨어 시뮬레이션

Encryption를 구현한 결과 그 크기가 하드웨어는 7,200Gates, 펌웨어는 2Kbyte임을 알 수 있었다. Encryption은 선택적으로 사용되는 부분으로서 생성된 암호화 스트림이 앞단의 결과와 XOR되어 전송하고자 하는 데이터를 암호화하게 된다. 따라서 Encryption이 선택되어지면 실시간으로 암호화 스트림이 생성되어야 하고, 구현 크기를 보더라도 하드웨어로 구현하는 것이 적당할 것이다. 이는 암호화기 알고리즘의 구성이 대부분 LFSR로 구성되어 있기 때문에 하드웨어 구현이 더욱 효율적일 수밖에 없다.

III. EISC 기반 통합설계

본 논문에서 설계하고자 하는 블루투스 칩의 내부 구성을 32비트 SE3208 코어, 블루투스 베이스밴드, 인터럽트, 메모리와 UART 블록으로 구성된다.

블루투스 베이스밴드 블록을 SE3208 코어와 연결하기 위해 중간자 역할을 하는 것이 버스 브릿지인데, 본 논문에서는 그림 10과 같이 버스 브릿지의 두 가지 모델을 제안하고 퍼포먼스가 좋은 방식을 제안하고자 한다. 그림10의 (a)는 블루투스 레지스터와 베이스밴드 블록을 CPU와 같은 버스 브릿지에 연결하여 제어하는 방식이고, (b)는 블루투스만의 버스 브릿지를 따로 두는, 즉 두개의 버스 브릿지로 제어하는 방식이다.

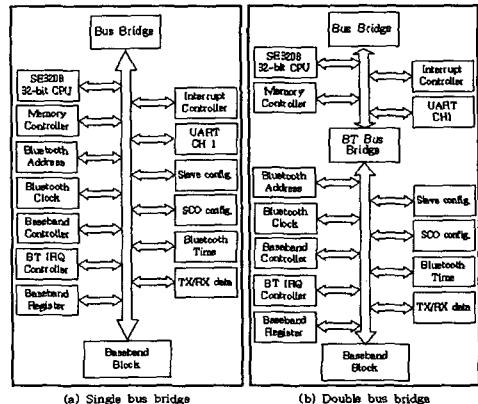


그림 10. 블루투스 코어 내부의 두 가지 모델

그림 10을 구현하여 검증하기 위해 그림 11의 검증 보드를 이용하였다. 이 보드는 CPU를 내장한 원 칩 블루투스 개발을 위해 외부회로가 구성되어있고, FPGA 디바이스는 Xilinx사의 XCV600HQ240C(약 660k Gate)을 사용하였다. 그리고 하드웨어 Verilog HDL 구현하였고, 펌웨어는 SE3208코어에 적합한 C로 구현하였다.

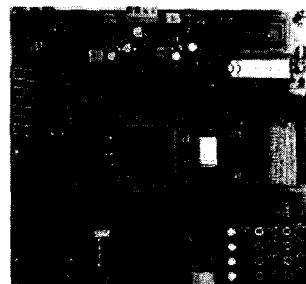
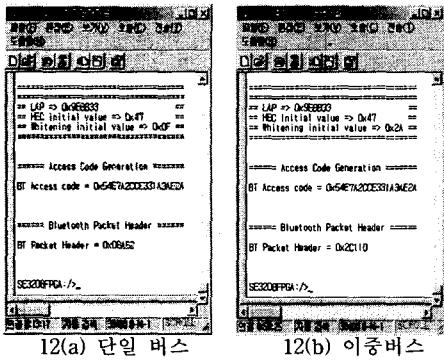


그림11. 검증 보드

본 논문에서 구현한 모델을 테스트하기 위한 방법은 다음과 같다. 우선 펌웨어를 통해 CPU에서 블루투스 레지스터에 필요한 값을 써주면 그 값을 토대로 블루투스 베이스밴드 블록에서는 그에 맞는 동작을하게 된다. 그리고 연산을 통해 얻어진 값은 다시 블루투스 레지스터에 저장되어 CPU에서 그 값을 읽어 들여 하이퍼터미널로 그 값을 출력하도록 하였다.

(1) 버스 브릿지 모델 구현 및 검증

그림 10(a)의 단일 버스 브릿지 모델로 패킷 생성을 구현한 결과는 그림 12(a)이고, (b)의 이중 버스 브릿지 모델로 패킷 생성을 구현한 결과는 그림 12(b)과 같다.



(2) 버스 브릿지 모델의 비교 및 검토

설계한 두 가지 버스 브릿지의 구현 결과, 그 크기가 단일 버스 브릿지인 경우 54,010Gates, 이중 버스 브릿지인 경우 54,133Gates이었고, 표 1과 같은 생성 클럭에 관한 정보를 얻을 수 있었다.

표 1. 생성 클럭

Clock Net	Max Skew(ns)			Max Delay(ns)		
	Single Bus bridge	Double Bus bridge	Ratio	Single Bus bridge	Double Bus bridge	Ratio
DIV_clk_b_1	2.481	2.101	1.181	9.141	9.217	0.991
DIV_clk_b_2	0.816	2.315	0.352	6.954	4.806	1.447
UART_CH0_rxclk	3.146	0.813	3.870	5.577	6.633	0.841
UART_CH0_tx16cnl<3>	2.485	2.009	1.237	4.657	4.518	1.031
UART_CH0_tx_empty	0.000	0.000	0	1.362	2.837	0.632
UART_CH0_txclk	0.595	1.072	0.555	4.269	3.416	1.250
SE3208_TOP_MAIN_CTRL_PI_PELINE_CTRL_qs<>	0.595	1.616	0.368	5.592	7.545	0.741
DIV_clk_b	3.408	1.098	3.104	6.435	7.518	0.845
BT_BB_extclk_reg_0_n0001	2.346	1.915	1.225	6.916	3.979	1.736
UART_CH0_txclk_sic	1.023	0.240	4.263	3.997	3.948	1.012
BT_BB_AC_COR_n0004	5.195	5.353	0.970	6.782	6.925	0.979
UART_CH0_tx_full	0.000	0.000	0	2.506	2.309	1.065
Average			1.43			1.05

구현 결과를 비교해 보면, 사용한 시스템 게이트 수는 단일 버스 브릿지가 더 유리한 결과를 얻었지만, 그 차이는 아주 미소하다. 하지만 생성 클럭 정보를 보면, 이중 버스 브릿지의 성능이 크게 우수함을 알 수 있다. 비록 SE3208 코어의 파이프라인 컨트롤 부분에서 단일 버스 브릿지가 유리한 결과가 나왔지만 블루투스 클럭과 UART에서 이중 버스 브릿지가 아주 유리한 결과가 나왔고 블루투스 동작의 정확성이 설계하고자 하는 블루투스 칩에서는 무엇보다 중요하다. 따라서 이중 버스 브릿지를 사용하는 것이 좀 더 안정적인 블루투스 시스템을 설계할 수 있을 것이다.

표 2. 결과 비교

요소 구조	클럭스워 크(%)	임박도달 시간(%)	출력지연 시간(%)	클럭주기 (%)	조합경로 지연(%)	LUT(%)	게이트 (%)
단일 버스 브릿지	100.0	100.0	100.0	100.0	100.0	99.6	99.9
이중 버스 브릿지	63.6	84.5	97.5	98.9	81.3	100.0	100.0
	여중	여중	여중	여중	여중	단일	단일

블루투스는 실시간으로 동작함으로 타이밍에 민감하며 타이밍이 어긋날 시 데이터를 유실한 가능성이 있다. 표 2에서 볼 수 있듯이 타이밍 측면에서는 이중 버스 브릿지가 유리하며 LUT와 게이트면에서는 단일 버스 브릿지가 유리하지만 차이가 그다지 크지 않기 때문에 이중 버스 브릿지로 구현해도 무방하도록 할 수 있다. 따라서 향상된 성능과 속도를 가진 블루투스 시스템을 설계하기 위해서도 이중 버스 브릿지로 설계하는 것이 효율적임을 알 수 있다.

IV. 결론

본 논문에서는 블루투스 베이스밴드의 전체적인 흐름을 이해하고, 각 블록의 알고리즘을 분석하였다. 그 후 베이스밴드 주요 블록을 구현하고, FPGA(Altera)와 MCU(Jupiter)를 원 보드에서 테스트하고 검증하였다. 블루투스 원칩 개발을 위하여 EISC Core, 내부 주변기기, 블루투스 베이스밴드를 하나로 연결하여 안정적이고 원하는 동작을 수행하는지 Xilinx 디바이스를 이용하여 검증하였다. 그리고 두 가지 방법으로 구현하여 얻어진 데이터를 서로 비교 검토하여 보다 효율적인 접합방식을 제안하고자 하였다. 그 결과 본 논문에서 제안한 두 가지 모델 중에 ‘double bus bridge’ 구조가 자연이나 구성요소의 사용면에서 유리하다는 것을 알 수 있었다.

본 논문은 블루투스 시스템의 성능을 향상시킬 수 있는 기초 자료로 활용될 것으로 사료되고, 구현된 칩은 소형, 저가를 선호하는 미래의 유비쿼터스 시장에서 매우 유용하게 사용될 것으로 생각된다.

참고 문헌

- [1] Jennifer Bray and Charles F STURMAN, 2001, BLUETOOTH Connect without cables, Prent Hall
- [2] Ciletti, Advanced Digital Design with the VERILOG HDL, 2002, Prent Hall
- [3] SIG, 2000, Specification of the Bluetooth System Version 1.1 Part B : Baseband specification, pp.41-178.